What is claimed is:

1. A multi-pass method of rendering a plurality of graphic primitives comprising:

in a first pass:

passing only a limited set of graphic data for each primitive through a graphic pipeline;

processing the limited set of data to build a compressed z-buffer, the compressed z-buffer comprising a plurality of z-records, each z-record embodying z information for a plurality of pixels;

setting a visibility indicator, for each primitive, if any pixel of the primitive is determined to be visible;

in a second pass:

for each primitive, determining whether the associated visibility indicator for that primitive is set;

discarding, without passing through the graphic pipeline, the primitives for which the associated visibility indicator is not set;

passing a full set of graphic data for each primitive determined to have the associated visibility indictor set; and

performing a two-level z-test on graphic data, wherein a first level of the z-test compares the graphic data of a current primitive with corresponding information in the compressed z-buffer, and wherein a second level of the z-test is performed on a per-pixel basis in a conventional z-test matter, wherein the second level z-test is performed only on pixels within a record of the compressed z-information in which the first level z-test determines that some but not all pixels of the macropixel are visible.

2. The method of claim 1, wherein passing only a limited set of graphic data more specifically comprises passing only location-related data through the pipeline.

3. The method of claim 2, wherein location-related data comprises X, Y, Z, and W values.

4. The method of claim 1, wherein each compressed z-record comprises a minimum z value for the plurality of pixels, a maximum z value for the plurality of pixels, and a coverage mask, the coverage mask indicating which of the plurality of pixels are visible for the current primitive.

5. The method of claim 1, wherein each compressed z-record comprises two minimum z values for the plurality of pixels, two maximum z values for the plurality of pixels, and a coverage mask, the coverage mask indicating which of the plurality of pixels are visible for the current primitive.

6. The method of claim 1, wherein setting the visibility indicator more specifically comprises setting a bit in a frame buffer memory.

7. The method of claim 1, wherein the discarding is performed by a parser.

8. A method of rendering a plurality of graphic primitives comprising:

processing, within a graphic pipeline, only a limited set of graphic data for each primitive;

determining, for each primitive, whether the primitive has at least one visible pixel;

processing, within the graphic pipeline, a full set of graphic data for only those primitives determined to have at least one visible pixel.

9. The method of claim 8, further comprising setting a visibility indicator for each pixel determined to have at least one visible pixel.

10. The method of claim 9, wherein setting the visibility indicator more specifically comprises setting a bit in a frame buffer memory.

11. The method of claim 8, wherein the processing only a limited set of graphic data more specifically comprises processing only location-related data.

12. The method of claim 8, wherein the determining whether the primitive has at least one visible pixel ensures that the primitive does not fail a compressed z-buffer test, ensures that all pixels of the primitive are not culled, ensures that the primitive does not render to zero pixels, and ensures that all pixels of the primitive are not clipped.

13. A method of rendering a plurality of graphic primitives comprising:

processing in a first pass, within a graphic pipeline, only a limited set of graphic data for each primitive;

processing the limited set of data to build a compressed z-buffer, the compressed z-buffer comprising a plurality of z-records, each z-record embodying z information for a plurality of pixels;

in a second pass, within the graphic pipeline, performing a two-level z-test on graphic data, wherein a first level of the z-test compares the graphic data of a current primitive with corresponding information in the compressed z-buffer, and wherein a second level of the z-test is performed on a per-pixel basis in a conventional z-test matter, wherein the second level z-test is performed only on pixels within a record of the compressed z-information in which the first level z-test determines that some but not all pixels of a macropixel are visible.

14. A graphics processor comprising:

first-pass logic configured to deliver to a graphic pipeline, in a first pass, only a limited set of graphic data for each primitive;

logic configured to process the limited set of graphic data for each primitive to create a compressed z-buffer;

logic configured to determine, for each primitive, whether the primitive has at least one visible pixel;

second-pass logic configured to deliver to the graphic pipeline, in a second pass, a full set of graphic data for only those primitives determined to have at least one visible pixel, the second-pass logic further configured to inhibit the delivery of

graphic data to the graphic pipeline for primitives not determined to have at least one visible pixel.

15. The graphics processor of claim 14, wherein the first-pass logic and second-pass logic are contained within a parser.

16. The graphics processor of claim 14, wherein the logic configured to determine whether the primitive has at least one visible pixel ensures that the primitive does not fail a compressed z-buffer test, ensures that all pixels of the primitive are not culled, ensures that the primitive does not render to zero pixels, and ensures that all pixels of the primitive are not clipped.

17. The graphics processor of claim 14, further including logic for setting a visibility indicator for each primitive determined to have at least one visible pixel.

18. The graphics processor of claim 17, wherein the visibility indicator includes a single bit in a frame-buffer memory.

19. The graphics processor of claim 17, further including logic configured to associate each primitive processed in the first pass of the data with a distinct visibility indicator.

20. The graphics processor of claim 19, further including logic configured to evaluate, for each primitive presented for processing in the second pass, a status of the visibility indicator associated with the given primitive.

21. A graphics processor comprising:

logic configured to limit the processing of graphic data for each of a plurality of primitives, in a first pass within a graphic pipeline, wherein the limited processing determines whether the primitive has at least one visible pixel;

logic configured to render, in a second pass within the graphic pipeline, each primitive determined in the first pass to have at least one visible pixel.

22. The graphics processor of claim 21, wherein the logic configured to limit the processing ensures that the primitive does not fail a compressed z-buffer test, ensures that all pixels of the primitive are not culled, ensures that the primitive does not render to zero pixels, and ensures that all pixels of the primitive are not clipped.

23. The graphics processor of claim 21, wherein the logic configured to limit the processing of graphic data is within a parser.

24. The graphics processor of claim 21, further including logic configured to build a compressed z-buffer of data from processing of the graphic data in the first pass.

25. The graphics processor of claim 21, further including logic for setting a visibility indicator for each primitive processed in the first pass.

26. The graphics processor of claim 21, further including logic configured to evaluate the visibility indicator for each primitive prior to submitting the primitive to the logic configured to render in the second pass.